Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.85
DOI: https://doi.org/10.24867/07DS06Somborac

PRIMENA MAŠINSKOG UČENJA U PROGNOZIRANJU

APPLICATION OF MACHINE LEARNING IN FORECASTING

Aleksandar Somborac, Bojan Jovanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast –SAOBRAĆAJ

Kratak sadržaj – *Zadatak ovog istraživanja jeste da se za potrebe prognoziranja izradi aplikacija koja vrši proračune koristeći savremene tehnike mašinskog učenja. Za izradu aplikacije korišćen je programski jezik Python zbog specifičnih prednosti koje on pruža u oblasti Nauke o Podacima (Data Science). Korišćeni su i razni softveri poput Git Bash, Anaconda, Jupyter Notebook i drugi, kao i razne biblioteke za matematičke operacije, za rad sa podacima, za crtanje grafika i za modele mašinskog učenja. Sama aplikacija omogućava prognoziranje indikatora razvoja država za jednu odabranu državu, za period od deset godina, na osnovu utiacaja odabrana dodatna tri indikatora na taj indikator. Uz aplikaciju prinuđen je i sifrarnik indikatora i država za korišćenje aplikacije.*

Ključne reči: *Python, Mašinsko učenje, Prognoziranje*

Abstract – *The goal of the research was to create an application for forecasting purposes using modern techniques of machine learning. Programming language called Python was used for the development of the application, specifically for it's advantages in Data Science. Other software like Git Bash, Anaconda, Jupyter Notebook and others were also used, as were Python libraries for mathematical operations, for data manipulation, for creating charts and for machine learning models. The application itself allows it's user to forecast data for a world development indicator, for a specific country, for a time period of ten years, based on the correlation of the additional three indicators on the main indicator. The application is acompanied by a codebook with all possible countries and indicators.*

Keywords: *Python, Machine learning, Forecasting*

1. UVOD

Ovo istraživanje je deo oblasti Data Science-a. Data Science (Nauka o podacima / Analiza podataka) je multidisciplinarna oblast koja koristi naučne metode, procese, algoritme i sisteme da pronađe nova znanja i uvide koristeći velike količine strukturiranih i nestrukturiranih podataka.
Pored termina Data Science, često se koriste i termini Data Mining (rudarenje podataka) ili Big Data (veliki podaci) [1].

Data Science koristi najmoćnije hardvere, najbolje programske jezike, modele mašinskog učenja i najefikasnije algoritme za rešavanje svakakvih problema. Kaže se i da je Data Science koncept koji treba da ujedini oblasti statistike, analize podataka i mašinskog učenja i njihove metode kako bi se došlo do najboljih rešenja.

Sa značajnim razvojem istovremeno i softvera i hardvera, računarski sistemi postaju sve moćniji, a sve je više podataka na raspolaganju. U velikoj količini podataka nalazi se i veliki broj rešenja i zaključaka, ali je izazov pronaći ih.

Iako je tehnički moguće pronaći iste zaključke koristeći sve programske jezike, Python je postao ubedljivi lider u oblasti Data Science – a. Data Scientst – teži da što pre i na što lakši i bolji način dođe do željenih zaključaka ili predviđana, a Python sa svojom lakoćom kucanja, učenja i čitanja i velikim brojem dostupnih biblioteka pruža upravo to. Omogućava programerima da se fokusiraju na direktan problem u vezi podataka, a da ne gube vreme sa problemima programiranja.

2. PROGRAMSKI JEZIK PYTHON

Python je interpretirani programski jezik visokog nivoa opšte namene. Osmišljen je sa filozofijom lakoće čitanja koda. Struktura ovog jezika i objektno orijentisani pristup omogućavaju programerima da pišu čist i logičan kod za male i velike projekte. Python je jezik dinamičkog tipa promenljivih. Pored toga što ga je lako podesiti i instalirati, često kaže da dolazi "sa baterijama", jer čak i osnovna verzija sadrži veliki broj biblioteka.

2.1. Istorija i razvoj Python-a

Python je koncipirao Guido van Rossum krajem 1980-ih na institutu Centrum Wiskunde & Informatica u Holandiji. Osmišljen je kao naslednik ABC jezika (zasnovan na SETL jeziku), sa mogućnošću da prepoznaje izuzetke i greške u kodu pre samog pokretanja koda. Implementacija jezika je započeta u decembru 1989. godine.

2.2. Filozofija jezika

Python kao jezik je zamišljen za veliki i različiti broj programera, gde će svaki imati svoj unikatan stil pisanja koda. Zasnovan je na principima objektno orijentisanog programiranja. Python je interaktivan i interpretiran programski jezik. Podržava i sve vrste strukturiranog, funkcionalnog i objektno orijentisanog programiranja.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio doc. dr Bojan Jovanović.

Sve promenljive su dinamičkog tipa, a jezik podržava i dinamičko rešavanje imena, to jest, može da poveže imena metode i promenljive prilikom izvršavanja koda.

Osnovni princip na kojem je programski jezik Python osmišljen i ostvaren je da kod bude pregledan i čitljiv. Većina drugih programskih jezika često nailazi na probleme gde programski kod bude previše apstraktan da bi se lako razumeo i pročitao.

Ovaj problem pogotovo ima veliki uticaj na početnike, a i na projekte gde sarađuje veliki broj ljudi. Na velikim projektima programeri konstantno moraju da čitaju, prerađuju i koriste kod svojih kolega, a ako je taj kod apstraktan i nerazumljiv na prvi pogled, to dovodi do usporenja rada i pojava grešaka.

2.3. Python biblioteke za Data Science

Python biblioteke koje se često koriste za Data Science:

- Numpy - dodaje podršku za rad sa velikim, multi-dimenzionalnim redovima i matricama. Uključuje i veliki broj matematičkih funkcija visokog nivoa za rad nad tim redovima i matricama.
- Pandas - za manipulaciju podacima i njihovu analizu. Zapravo obezbeđuje određene strukture podataka i operacije za manipulisanje numeričkim tabelama. Ime biblioteke je zapravo skraćenica dve reči Panel Data, što znači tabelarni podaci.
- Matplotlib - omogućava crtanje 2D i 3D grafika. Podržava crtanje grafika i za osnovne Python liste, a podržava i NumPy matrice i Pandas DataFrame tabele. Matplotlib ima i aplikaciono programski interfejs (API) za rad sa grafičkim alatima za izradu Python aplikacija, to jest, grafici mogu lako da se integrišu u Desktop aplikacije.
- Scikit – learn - biblioteka za mašinsko učenje u programskom jeziku Python. Omogućava algoritme klasifikacije, regresije i klasterizacije i napravljena je kao nadogradnja biblioteka NumPy i SciPy.
- TkInter - najbolji izbor za RAD GUI (Rapid Application Development, Graphical User Interface), to jest, za brzi razvoj desktop aplikacija sa grafičkim korisničkim interfejsom. Omogućava razvijanje prozora za aplikacije koji su robustni i nezavisni od platforme na kojoj se nalaze.

3. KORIŠĆENI SOFTVERI

Za izradu aplikacije korišćeni su sledeći softveri:

Git Bash, Anaconda, Jupyter Notebook i drugi.

Git Bash

Jezgro Git-a čine kolekcija komandi i korisnih programa koji mogu da se izvršavaju preko command-lineenviromenta (okruženje za za izvršavanje koda dužine jednog reda), kao što je na primer, Microsoft Windows-ov Command Prompt.

Anaconda

Anaconda je besplatna i open-source distribucija Python i R programskih jezika za specifične primene i nauke kao što su Data Science, mašinsko učenje, procesiranje velike količine podataka, analize sa prediktivnim sposobnostima i slično. Cilj Anakonde je da pojednostavi upravljanje, instalaciju i pokretanje raznih okruženja. Sva okruženja i paketa koja pruža pokreću se posebnom Bash komandom conda.

Jupyter Notebook

Jupyter Notebook je interaktivno razvojno okruženje kojem se pristupa preko web pretraživača. Specifičnost ovog razvojnog okruženja, za razliku od mnogih, je to što se kod u jednoj datoteci može sadržati u više nezavisnih Notebook-ova, ili svesaka. Svaka sveska ima svoj redni broj, a taj redni broj predstavlja redni broj izvršavanja svih svesaka. Iako su sve sveske nezavisne, to jest, mogu da se pokreću pojedinačno i nezavisno od redosleda, one i dalje dele sve globalne promenljive u toj datoteci.

Ostali softveri korišćeni u izradi projekta:

- Spyder – razvojno okruženje
- Visual Studio Code – tekst editor za pregled koda
- Python tutor – vizuelizacija izvršavanja koda
- py2exe – za kreiranje .exe datoteke

4. MAŠINSKO UČENJE

Mašinsko učenje je naučna oblast koja obuhvata proučavanje algoritama i statističkih modela koje koriste računarski sistemi za izvršavanje nekih zadataka [2]. Za izvršavanje takvih komplesnih zadataka ne koriste se eksplicitne komande i instrukcije, već se sitemi oslanjaju na šablone i korake zaključivanja. Oblast mašinskog učenja se smatra podgrupom veštačke inteligencije i predstavlja osnovni alat kojim veštačka inteligencija dolazi do zaključaka i dolazi do razumevanja o svojoj okolini.

Algoritmi mašinskog učenja izgrađuju matematičke modele na osnovu uzoraka, to jest podataka za "treniranje". Na osnovu tih podataka, dolaze do određenih zaključaka i odluka na osnovu kojih mogu da vrše predviđanja ili izvršavanja nekih zadataka bez eksplicitnih naredbi. Algoritmi mašinskog učenja koriste se za široki spektar raznih aplikacija, kao što su filtriranje e-mailova, računarska vizija (računarsko razumevanja slika i videa), predviđanja podataka, grupisanje podataka i slično.

4.1. Tipovi algoritama mašinskog učenja

Osnovna podela tipova mašinskog učenja je po tome da li postoji nadzora pri učenju ili ne. Tipovi algoritama mašinskog učenja su:

- Mašinsko učenje pod nadzorom,
- Mašinsko učenje bez nadzora,
- Pojačano učenje.

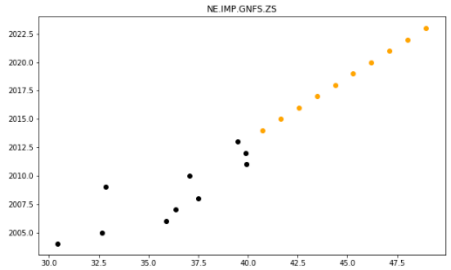
Algoritmi mašinskog učenja pod nadzorom prave matematičke modele na osnovu podataka koji sadrže i ulaze i izlaze, to jest baziraju se na osnovi podataka za treniranje. Koriste primere i podatke iz prošlosti na kojima pronalaze određene šablone i zaključke kako bi mogli to znanje preneti na podatke u budućnosti.

Mašinsko učenje bez nadzora je tip samo-organizovanog učenja koji pronalazi prethodno nepoznate šablone i zakonitosti nad nekim podacima. Dve osnovne metode ovakvog učenja su analiza klastera i pronalaženje glavne komponente. Za pronalaženje glavne komponente vrši se ortogonalna transformacija podataka sa gausovom raspodelom, a rezultat je par vrednosti koji najbolje opisuje neki skup podataka. Najpoznatija metoda mašinskog učenja bez nadzora je analiza klastera.

Pojačano učenje sa zasniva na tome da programi treba da odabiraju da izvrše onakve akcije koje bi maksimizovale neki rezultat ili im osojile neku nagradu. Nalik treniranju kućnih ljubimaca tako što se poslasticom nagrađuje dobro ponašanje ili pravilno odrađen trik, računarski programi zasnovani na pojačanom učenju se motivišu kada izvrše pravu akciju.

4.2. Algoritmi regresije

Za prognoziranje u okviru aplikacije korišćeni su algoritmi regresije koji predstavljaju tip mašinskog učenja pod nadzorom. Specifično je korišćen deo biblioteke Scikit – learn pod imenom LinearRegression, to jest algoritam linearne regresije. Linearna regresija je tip regresione analize gde postoji jedna nezavisna promenljiva i postoji linearan odnos između nezavisne X i zavisne promenljive Y [3]. Na sledećoj slici, narandžaste tačkice predstavljaju pravu liniju koja najbolje opisuje odnos između dveju promenljive. Ta prava može da se modeluje na osnovu formule y = kx + n.



Slika 1. *Linearna regresija, izvor: autor*

Troškovna funkcija

Da bi se odredila optimalna linija regresije, potrebno je odrediti vrednosti za k i n za koje će linija regresije najbolje opisati odnos između dve promenljive. Pošto želimo da pronađemo najbolje vrednosti za k i n, ovaj problem je potrebno pretvoriti u problem minimizacije gde želimo da minimizujemo grešku između predviđene vrednosti i stvarne vrednosti. Sledeća formula se naziva troškovna funkcija (1):

$$R = \frac{1}{N}\sum_{i=1}^{N}(pred_i - y_i)^2 \qquad (1)$$

Silazni gradijent

U teoretskom primeru, troškovna funkcija prolazi kroz sve podatke i pronalazi funkciju sa minimalnom greškom [4]. U realnim slučajevima ipak dolaze do izražaja i dodatni faktori. Pošto se algoritmi mašinskog učenja najčešće primenjuju na ogromnoj koiličini podataka, performanse algoritma imaju značajnu ulogu u kvalitetu algoritma. Potrebno je ne samo pronaći funkciju sa minimalnom greškom, već to uraditi za najkraće moguće vreme.

Metoda koja se koristi za poboljšavanje performansi algoritma mašinskog učenja za linearnu regresiju se naziva silazni gradijent. Ova metoda ažurira vrednosti promenljivih k i n dok pronalazi minimalnu vrednost greške. Ideja je da metoda započne sa nekim vrednostima k i n i da ih iterativno menja dok pronalazi optimalnu funkciju.

Umesto da metoda prolazi kroz svaki podatak posebno, kojih može biti milioni, ova metoda preskače velike količine podataka, a one koje odabere posmatra kao uzorke na kojima testira vrednost greške. Pošto vrednost greške predstavljena na grafiku najčešće izgleda kao rupa, metoda koja silazi do dna je tako dobila ime silazni gradijent.

5. PROJEKAT IZRADE APLIKACIJE

Projekat u prilogu rada se sastoji od aplikacije napisane na programskom jeziku Python, koja korisnicima omogućava da vrše predviđana raznih indikatora uspeha raznih država koristeći modele i biblioteke mašinskog učenja. Aplikacija nudi odabir tri indikatora koje služe kao podaci za treniranje mašinskog učenja, odabir države za koju se predviđanje radi i odabir glavnog indikatora čije predviđenje vrednosti se određuju na osnovu uticaja tri odabrana indikatora.

Aplikacija kroz jedan ciklus izvršavanja koda zapravo obavlja proces Data Science-a, koji čine:

1. Određivanje problema
2. Prikupljanje podataka za analizu
3. Procesiranje i čišćenje podataka
4. Analiza i obrada podataka
5. Prezentacija rezultata

U slučaju datog projekta, problem se sastoji od predviđanja budućih vrednosti u zavisnosti od intenziteta uticaja odabranih indikatora na glavni indikator.

Podaci za analizu prikupljeni su sa web sajta www.Kaggle.com. Kaggle je je mesto na kojem Data-Scientist-i kače razne baze podataka pristupne svima, kao i ideje za procesiranje, analizu i prezentaciju tih podataka. Za potrebe projekta korišćena je baza podataka pod nazivom "World Development Indicators".

Pre nego što se započne bilo kakva analiza potrebno je očistiti bazu podataka koja se koristi i odabrati određeni

obim čistih podataka. Čišćenje podataka vršeno je filtriranjem podataka pod određenim uslovima, tako da se za odabrane države, godine i indikatore uvek nalazi samo jedan podatak, ni manje ni više. Države i indikatori čiji su podaci čisti, smešteni su u šifrarnik država i indikatora, kojima se može pristupiti preko aplikacije.

Analiza i obrada podataka otpočinje učitavanjem baze podataka, a zatim učitavanjem čistih podataka pomoću funkcija load za glavni indikator i load_all za dodatna tri indikatora. Dobijeni objekti, to jest, liste podakata za glavni indikator se naziva df (od reči Data Frame), a za ostale indikatore df_all. Oni se zatim smeštaju na model mašinskog učenja, na sledeći način:

```
regressor = LinearRegression ()
regressor.fit (df_all, df)
```

Zatim se za svaki od tri pomoćna indikatora vrši algoritam linearne regresije za prognoziranje podataka za period od deset godina, računa se prosećna kvadrirana greška i korelacija tog indikatora na glavni indikator, na sledeći način:

```
ind_regressor=LinearRegression().fit(x,y)
rmse = ind_regressor.score(x,y)

corr = ind_regressor.coef_[df]
```

Na osnovu dobijenih vrednosti, pomoću troškovne funkcije i algoritma silaznog gradijenta se određuju optimalne linije regresije. Zatim na osnovu uticaja svakog od tri indikatora na glavni indikator i na osnovu optimizovanih linija regresije, vrši se predviđanje glavnog indikatora, na sledeći način:

`z = regressor.predict (pred),` gde su:

z – lista od deset predviđenih vrednosti za glavni indikator, a pred je rečnik svih unapred pripremljenih vrednosti. To jest vršimo predviđanje na osnovu vrednosti pred koristeći model mašinskog učenja regressor.

Nakon što su sve vrednosti izračunate i predviđanja izvršena, potrebno je dostaviti rešenja korisniku. Za svaki od tri indikatora poziva se funkcija draw koja koristi alate biblioteke Matplotlib da iscrta grafike sa parovima vrednosti starih i predviđenih, gde su stare plave boje a predviđene vrednosti crvene boje. Posle finalnog predviđanja, na isti način se iscrta i grafik parova vrednosti glavnog indikatora, gde su stare vrednosti crne boje, a predviđene vrednosti narandžaste boje.

Osim grafičkog prikaza, predviđene numeričke vrednosti glavnog indikatora smeštaju se u datoteku .json formata pod imenom Results.

Osim samog postupka predviđanja, u projektu su korišćeni jos i alati biblioteke TkInter za grafički prikaz aplikacije, koraci za proveru unosa i ispisivanje grešaka, funkcije za uništavanje, to jest, gašenje aplikacije i funkcije za interakciju sa operativnim sistemom biblioteke OS, koje su omogućile otvaranje Excel datoteke sa prinuđenim šifrarnikom država i indikatora za korišćenje aplikacije.

6. ZAKLJUČAK

Nauka o podacima (Data Science), odnosno data science postaje sve popularnije ali i nejasno opširno zanimanje. Nije u pitanju samo kombinacija programiranja i statistike, već i poznavanje tematike koja se istražuje kroz podatke, kao i uspešno komuniciranje analize tih podataka kroz izveštaje i vizualizacije.
Sa sve većim rastom količine podataka u svetu i sa sve naprednijim i sve sofisticiranijim programskim jezicima i alatima, istraživanje podataka postalo je jedno od ključnih zanimanja u svetu.

U oblasti Data Science-a sve važnija postaje primena mašinskog učenja. Moderni modeli mašinskog učenja prevazilaze mogućnosti pisanja algoritama i najtalentovanijih programera i naučniika. Mašinsko učenje pronalazi primenu u teoriji igara, kontrole, informacija, u simulacijama, operacionim istraživanjima, stastistici i predviđanju, u vožnji autonomnih vozila, igranje igara protiv ljudskih protivnika, prepoznavanju slike, navigaciji terena i slično.

7. LITERATURA

[1] https://www.quora.com/What-is-data-science

[2] https://expertsystem.com/machine-learning-definition/

[3] https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a

[4] https://realpython.com/linear-regression-in-python/

Kratka biografija:

Aleksandar Somborac rođen je u Bačkoj Palanci 1995. god. Master rad na Fakultetu tehničkih nauka iz oblasti Poštanski saobraćaj i telekomunikacije odbranio je 2019.god.

kontakt: asomborac@gmail.com

Bojan Jovanović rođen je u Šapcu 1983. godine. Doktorirao je na Fakultetu tehničkih nauka Univerziteta u Novom Sadu 2015. godine.